CSC2125 Homework 3

Zi Han Zhao

1001103708

Part1

1 Suppose we run PBFT algorithm on a cluster of 21 machines. What is the maximum number of machines we will be able to tolerant for failure simultaneously?

We know $N \geqslant 3f+1$ where N is the total number of nodes and f replicas are faulty. Then $21 \geqslant 3f+1$ so $f_{max} = 6$.

2 A node in PBFT waits for prepare/commit messages from two thirds of nodes (replicas) during the prepare/commit stages. What would happen if the node instead waits for prepare and commit messages from only a simple majority of other nodes? Would the modified PBFT be secure? If not, please explain with a counter example.

The modified PBFT is not secure anymore. Let's say the number of responses of node A is x where $x < 2f+1$, f is the number of faulty nodes. Among the x messages, in the worst case, there would be possibly f faulty responses so there are $x-f$ non-faulty responses. It is found that $x-f < f+1$, which means that non-faulty responses is not greater than faulty responses. Therefore, node A cannot decide the message he received is correct or not.

3 PBFT relies on the primary node to send out pre-prepare messages to drive the consensus process. What would happen if the primary node is malicious or fails? How does PBFT handle this situation?

First, even if the primary node is malicious or fails, the whole system can still keep in consensus. The reason is that even if the primary node is malicious and sends different pre-prepare messages to each backup, each replica will receive recursive feedbacks from each other during prepare and commit stages, then some of replicas will notice the unusual action by primary. Next view-change is triggered and new primary node is selected.

There are 4 situations triggering view change:

- Primary node timeouts for sending message.
- Primary node send different message for each backup.
- Primary node never send pre-prepare message for client. Then client will broadcast messages to all replicas if the reply timeouts.

- Primary node overwrites messages. Other replicas will verify the message and trigger view-change.

A new primary $p' = (v+1) mod |R|$ will take effect if at least $2f+1$ replicas trigger view-change. Then the primary sends new view messages to each backup.

4 Could PBFT skip the commit messages? Suppose nodes in PBFT commit-local directly after receiving 2f + 1 prepare messages. What could go wrong? Please explain with a counter example (Consider the case where a view change happens immediately after one or two nodes commit-locally for a request).

No. PBFT cannot skip commit. If view change happens right after node A committed locally, node A cannot prove over $2f$ replicas receive the true messages and cannot keep in consensus. Then the log which node A stored locally is not valid and in consensus. Therefore, during view change stage, node A cannot recover the request which has reached before new primary is selected.

5 A permissioned blockchain is a blockchain system in which the membership of the blockchain is fixed and known to all participants. The membership may be managed offline by a potentially centralized organization. Explain how PBFT could be used to implement such a permissioned blockchain.

Implementation:

- Block proposal: some node claims a new block proposal and send pre-prepare message.
- Pre-prepare: validators receive pre-prepare messages and broadcast prepare message.
- Prepare: Upon receiving $2f+1$ prepares, validators send commit messages.
- Commit: Upon receiving $2f+1$ commits, validators insert new block into blockchain.
- Proposer change: Send proposer-change message. Upon receiving $2f+1$ messages, change a new proposer to claim new block.

Why only applying on permissioned blockchain? First, the recursive communication among replicas increases the time complexity. Such $O(n^2)$ algorithm cannot be applied on large-scale public chain. Second, the number of faulty nodes are easily estimated by centralized organization and the PBFT algorithm safety and liveness on blockchain will be evaluated.

6 Suppose the number of replicas in the system is N. During the normal-case operation in PBFT, roughly how many messages will be broadcasted for each request? Suppose the broadcast is implemented via point to point transmission between nodes one by one. What is the total number of transmitted messages for processing each request? (Answer in Big O notation like O(1), O(N), $O(N^2)$, ...)

$O(N^2)$. Because each node should broadcast messages to other $N-1$ nodes. There are $N*(N-1)$ communications in total. So it is $O(N^2)$.

7 Suppose the number of replicas in the system is N. Suppose the broadcast is implemented via point to point transmission between nodes one by one. What is the total number of transmitted messages during a view-change in PBFT? What is the size of each message during a view-change? (Answer in Bit O notations like O(1), O(N), $O(N^2)$, . . .)

Time complexity is $O(N^2)$ and the size of a single message is $O(N)$. Because during view-change, some nodes which notice unusual actions of primary broadcast view-change messages, which is $O(N^2)$. Then the new primary receives view-change messages and collect them, then broadcasts new-view messages, which is $O(N)$. Therefore, time complexity is $O(N^2)$.

The view-change message is $< VIEWCHANGE, v + 1, n, C, P, i >$. The size of view-change message consists of message title ($O(1)$), view number ($O(1)$), stable checkpoint ID ($O(1)$), a set of $2f + 1$ checkpoint messages ($O(N)$), a set of the requests' pre-prepare and $2f$ prepare messages and a sequence number ($O(N)$). Therefore the size is $O(N)$.

8 Given the above analysis, explain why it might be impractical to use PBFT to implement a large scale blockchain.

- First, the time complexity is $O(N^2)$, which lags two much during the communication under large scale blockchain. It is too inefficient.
- Second, current POW consensus protocol fault tolerance is 50% but PBFT is 33.3%. It means, under PBFT, it is easier to be attacked when the number of faulty nodes over 33.3%.
- Third, PBFT needs primary to control message communications. The degree of decentralization is lower than POW.
- The logs of transactions will be deleted after committing. It is easier for attackers to overwrite transaction history.

Part2

9 Why withholding a mined block can enable the attacker to gain more rewards? Note that the attacker still expects to generate the same number of blocks.